First paragraph on page 12:

A3

year, or model. The interaction sequence in this example involves [the] the choice of 2001 for year, in support of the user's goals.

Last paragraph on page 12:

Since PIPE only emphasizes the design and implementation of personalization systems, it does not pay any attention to how the interaction sequences are obtained and how the choice between terminal and structural parts is made. In particular, PIPE is not a complete life cycle model for personalization system design and does not address issues such as requirements gathering, Interaction sequences could come from explaining users' behavior (see, for example, N.Ramakrishnan, M.B. Rosson, and J.M. Carroll, "Explaining Scenarios for Information Personalization, <u>communicated to</u> *ACM Transactions on Computer-Human Interction,* August 2001, and A.Wexelblat and P. Maes, "Footprints: History-Rich Tools for information Foraging, *Proceedings of the ACM Conference on Human Factors in Computing Systems (CHI'99)*, pp. 270-277, Pittsburgh,

A4

Second paragraph on page 18:

The naive rendition of a PIPE model by the above mechanisms might result in lengthy programs, with duplication of interaction sequences. Techniques for program compaction of interaction topic has been studied extensively in the data mining and [semistructored] <u>semistructured</u> modeling communities (see, for example, S. Abiteboul, P.Buneman, and D. Sucie, *Data on the Web: From Realtions to Semistructured Data and XML*, Morgan Kaufman Publishers, 2000, S. Nestorov, S. Abiteboul, and R. Motwani, "Extracting Schema from Semistructured Data", *Proceedings of the ACM International Conference on Management of Data (SIGMOD'98)*, pp.165-176, 1988, and K. Wang and H. Liu, "Discovering Structural Association of Semicostructured Data", *IEEE Transactions on Knowledge and Data Engineering*, Vol. 12(3): pp. 353-371, May/June 2000). Of particular relevance to PIPE is the algorithm of Nestorov et al., *supra*, whose modeling of semistructure closely resembles the representation of an interaction of an interaction sequence in terms of program variables. This algorithm works by identifying graph constructs that could be factored, or approximated. Figures 9A, 9B, 9C, and 9D illustrate four stages in a procedure for program compaction. The starting

A5 Cm't

A5 Cncld.

point is the schema in Figure 9A obtained by a naive crawl of a site. Figure 9B factors commonalities encountered in crawling. There are only three leaf nodes and the internal nodes P3 and P4 are collapsed because they are really the same page. Figure 9C is a "minimal perfect typing" (from Nestorov et al., supra) of the data, which means that the fewest internal

The last paragraph on the page 19:

A6

There are now described two applications that use PIPE to personalize collections of web sites. They are presented in increasing order of complexity, as evidenced by the forms of modeling they conduct (see Table 1). In each of these applications, the conceptual model of interaction sequences and the specific choices made in modeling are stated. Evaluation results are presented in N. Ramakrishnan and S. Perugini, "The Partial Evaluation Approach to Information Personalization", Technical Report cs.IR/0108003, Computing Research Repository (CoRR), August 2001. Since PIPE only specializes representations, it is possible to personalize even third-party sites by forming suitable representations. More personalization systems designed with PIPE are described in N. Ramakrishnan, "PIPE: Web Personalization by Partial Evaluation", IEEE Internet Computing, Vol. 4(6): pp.21-31, Nov-Dec 2000, and N. Ramakrishnan, M. B. Rosson, and J. M. Carroll, "Explaining Scenarios for Information Personalization" communicated to *ACM Transactions on Computer-Human Interaction*, August 2001. Only two are presented here for space considerations.

The last paragraph on page 23:

A7 Cm.t

The entire GAMS web site was modeled, the PYTHIA recommender (that addressed software for the domain of PDEs) was used, and connections with individual software modules at the various repositories were established. After an initial expansion of GAMS (e.g., by within-page modeling), the program compaction algorithm described [in above] earlier was applied. Cross-references in GAMS and duplication of common module sets (which are now revealed by the initial expansion) helped compress the site schema to sixty percent of its original size. [rn] In particular, the GAMS subtree relevant to describing PDEs provided for a eleven percent compression. There was no terminal information alongside intermediate nodes, and hence there was no need for any special handling. PYTHIA's details are described in E.N. Houstis et al., *supra*, and a white-box modeling in PIPE was conducted to better associate program variables

from GAMS with variables in PYTHA. Finally, the step to reach individual software modules was a simple one-step interaction sequence leading to terminal information about the code (in FORTRAN) and its documentation. The entire composite program was represented in the CLIPS programming language (see J.C. Giarratano, *Expert Systems: Principles and Programming*, Brooks/COle Publishing, 1998) and its rule-based interface for partial evaluation was employed. More modeling details on this case

A7 Cont'd.

The last paragraph on page 27:

A8

The next step in the PIPE methodology/process shown in Figure [14] 11 is compacting interaction sequences. This step is optional. Its purpose is to compact the set of interaction sequences to determine a new set of

First paragraph on page 28:

A9

interaction sequences that has fewer states. Any standard algorithm for state minimization or schema compression can be utilized for this step. One way to do this is the procedure shown in Figure [10] 9.

The third paragraph on page 28:

A10

The next step in the PIPE methodology/process shown in Figure [14] 11 is the programmatic representation of interactions sequences. This step involves the representation of the set of interaction sequences as a computer program. The programming language this serves as the representation language. Any imperative programming language can be use, such as C, Pascal, FORTRAN, and BASIC.

The third paragraph on page 29:

A11

The next step in the PIPE methodology/process shown in Figure [14] 11 is creating a personalization system. This step takes the computer program created in the previous step and uses it to form the basis of a presentation system. This step uses a partial evaluator and an information space generator to complete the system design.